TUGAS AKHIR – TE 145561

SISTEM *LOCK DOOR* PANEL LISTRIK MENGGUNAKAN *FINGERPRINT SCANNER* BERBASIS ANDROID

Aswindha Nasrullah
NRP 2214039011

Dosen Pembimbing
Rachmad Setiawan, ST., MT.
Subadi, S.Pd.

PROGRAM STUDI ELEKTRONIKA INDUSTRI
Departemen Teknik Elektro Otomasi
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

FINAL PROJECT – TE 145561

LOCK DOOR ELECTRIC PANEL SYSTEM USING FINGERPRINT SCANNER BASED ON ANDROID

Aswindha Nasrullah
NRP 2214039011

Advisor I
Rachmad Setiawan, ST., MT.

Advisor II
Subadi, S.Pd.

ELECTRICAL INDUSTRY STUDY PROGRAM
Electrical and Automation Engineering Department
Vocational Faculty
Institut Teknologi Sepuluh Nopember
Surabaya 2017

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "**SISTEM *LOCK DOOR* PANEL LISTRIK MENGGUNAKAN *FINGERPRINT SCANNER* BERBASIS ANDROID**" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diijinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 20 Juli 2017

Aswindha Nasrullah
NRP. 2214039011

-----Halaman ini sengaja dikosongkan-----

SISTEM LOCK DOOR PANEL LISTRIK MENGGUNAKAN FINGERPRINT SCANNER BERBASIS ANDROID

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Ahli Madya
Pada
Program Studi Elektronika Industri
Departemen Teknik Elektro Otomasi
Fakultas Vokasi
Institut Teknologi Sepuluh Nopember

Menyetujui:

Dosen Pembimbing I	Dosen Pembimbing II
Rachmad Setiawan, ST., MT.	Subadi S.Pd.
NIP. 19690529 199512 1 001	NIP. 19630923 198603 1 018

SURABAYA
JULI, 2017

-----Halaman ini sengaja dikosongkan-----

SISTEM *LOCK DOOR* PANEL LISTRIK MENGGUNAKAN *FINGERPRINT SCANNER* BERBASIS ANDROID

Nama Mahasiswa	**:**	**Aswindha Nasrullah**
NRP	**:**	**2214 039 011**
Dosen Pembimbing I	**:**	**Rachmad Setiawan, ST., MT**
NIP	**:**	**19690529 199512 1 001**
Dosen Pembimbing II	**:**	**Subadi, S.Pd**
NIP	**:**	**19630923 198603 1 018**

ABSTRAK

Pada tugas akhir ini membahas masalah sistem *lock door* pada panel listrik menggunakan sensor *fingerprint scanner* yang praktis dan dapat berperan sebagai *data logger* untuk mencegah tindak kriminalitas yang dilakukan selama pengoperasian pintu panel. Sistem ini memberikan informasi kepada pengguna tentang kapan dan ID pengguna siapa yang membuka pintu panel.

Sistem *lock door* dirancang menggunakan mikrokontroller, *fingerprint scanner* sebagai sensor utama untuk mendeteksi kecocokan data sidik jari pengguna, modul *bluetooth* HC-05 sebagai media komunikasi antara alat serta Android. Sedangkan untuk *software* menggunakan IDE untuk Arduino sebagai pemrograman inti dan *software* MIT APP Inventor 2 sebagai *interfacing*. Hasil simulasi serta implementasi menunjukkan bahwa sistem *lock door* yang telah dirancang memiliki sensitivitas yang tinggi dan mampu melakukan perannya sebagai data logger melalui kabel serial.

Kata Kunci : *Fingerprint,* Arduino Mega 2560, Android, *Bluetooth HC-05*.

-----Halaman ini sengaja dikosongkan-----

LOCK DOOR ELECTRIC PANEL SYSTEM USING FINGERPRINT SCANNER BASED ON ANDROID

Student Name	**:**	**Aswindha Nasrullah**
ID Number	**:**	**2214 039 011**
***Advisor* I**	**:**	**Rachmad Setiawan, ST., MT**
ID Number	**:**	**19690529 199512 1 001**
***Advisor* II**	**:**	**Agus Suhanto, S.Pd**
ID Number	**:**	**19630923 198603 1 018**

ABSTRACT

This final project disscus about the problem of lock door system on electric panel using fingerprint scanner that can act as a data logger to prevent crime activity that will be do by person that we used to know during the operation of door panel. This system provides users with information about when and whose user ID open the panel door. This sistem provides information to user when and who the users are access the door.

Lock door system is designed using microcontroller and fingerprint scanner as main sensor to detect matches of user's fingerprint. Communication system using bluetooth HC - 05 and Android. The simulation result and the implementation showed that the door lock system that designed has a high sensitivity and able to perform it's role as data logger.

Keywords : Fingerprint, Arduino Mega 2560*,* Android*, Bluetooth HC - 05*

-----Halaman ini sengaja dikosongkan-----

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT yang selalu memberikan rahmat dan hidayah-Nya sehingga Tugas Akhir ini dapat terselesaikan dengan baik. Shalawat serta salam semoga selalu dilimpahkan kepada Rasulullah Muhammad SAW, keluarga, sahabat dan umat muslim yang senantiasa meneladani beliau.

Tugas Akhir ini disusun untuk memenuhi sebagian persyaratan guna menyelesaikan pendidikan Diploma pada Program Studi Elektro Industri, Departemen Teknik Elektro Otomasi, Fakultas Vokasi, Institut Teknologi Sepuluh Nopember Surabaya dengan judul:

"SISTEM LOCK DOOR PANEL LISTRIK MENGGUNAKAN FINGERPRINT SCANNER BERBASIS ANDROID"

Dalam Tugas Akhir ini dirancang sebuah sistem keamanan pintu panel menggunakan *fingerprint scanner* sebagai *user detector* agar pemilik ID bisa membuka pintu dan adanya *data logger* sebagai aplikasi pendukung sistem *lock door*.

Dengan terselesaikannya Tugas Akhir ini, penulis menyampaikan terima kasih yang sebesar - besarnya kepada :

1. Kedua orang tua dan keluarga yang senantiasa mendoakan dan memberikan dukungan dengan tulus tiada henti.
2. Bapak Rachmad Setiawan, ST., MT. selaku dosen pembimbing.
3. Bapak Subadi, S.Pd. selaku dosen pembimbing dari BLKIP.
4. Teman-teman Elektro Industri 18 dan Andromeda DE-09 yang selalu memberikan doa, semangat serta dukungan.
5. Semua pihak yang telah membantu baik secara langsung maupun tidak langsung dalam proses penyelesaian Tugas Akhir ini.

Penulis menyadari dan memohon maaf atas segala kekurangan pada Tugas Akhir ini. Akhir kata, semoga Tugas Akhir ini dapat bermanfaat dalam pengembangan ilmu di kemudian hari.

Surabaya, Juli 2017

Penulis

-----Halaman ini sengaja dikosongkan-----

DAFTAR ISI

DAFTAR GAMBAR

-----Halaman ini sengaja dikosongkan-----

DAFTAR TABEL

-----Halaman ini sengaja dikosongkan-----

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perusahaan PT. PLN (Persero) memiliki sistem tenaga listrik yang bersumber dari gardu induk (GI) serta melalui gardu grafo tiang (GTT) yang digunakan sebagai penghubung jaringan listrik ke konsumen jaringan tegangan menengah (20KV) dan konsumen tegangan rendah (220V/380V). Faktor keamanan merupakan hal yang diutamakan oleh PT. PLN (Persero). Sistem keamanan pintu panel dengan menggunakan kunci konvensional dianggap kurang praktis. Petugas PLN maupun vendor yang sedang melakukan *maintenance* terkadang lalai untuk membawa kunci panel.

Tindak kriminalitas dalam hal pencurian listrik telah banyak merugikan PT. PLN (Persero). Kasus pencurian yang diungkap pada PT. PLN (Persero) Distribusi Bogor yaitu pada perusahaan yang bergerak di bidang pemintalan benang dengan modus pencurian yaitu membobol gardu distribusi dan memutus kabel suplai arus yang kemudian disambungkan pada kabel lainnya, sehingga arus listrik tidak masuk ke KWh meter. Modus lain berupa pencurian material gardu distribusi seperti pada Desa Parit Lintang Kalimas, Kecamatan Sei Kakap, dengan cara membuka paksa gardu listrik milik PLN. Sebagai barang bukti telah ditemukan sebuah busbar *phase* N yang dilepas dari gardu nomor KP 0021, sisa barang bukti dua buah NH *fuse* yang tidak dapat diidentifikasi.

Untuk menanggulangi permasalahan di atas maka diperlukan sistem penguncian yang modern dan efisien untuk menciptakan kondisi aman pada panel distribusi yang dibandingkan dengan pengaman konvensional. Pengaman kunci pintu modern saat ini yang bisa diaplikasikan adalah penggunaan *fingerprint scanner.*

Oleh karena itu pada Tugas Akhir ini dibuat sistem *lock door* pintu panel menggunakan sensor *fingerprint scanner* dengan data akses pintu yang dapat dicatat dan ditampilkan pada PC dengan berbasis Android. Sehingga dapat meminimalisir tindak kriminal yang dilakukan bahkan oleh orang dalam.

1.2 Permasalahan

Permasalahan pada Tugas Akhir ini mengenai belum adanya sistem penguncian panel modern yang berperan sebagai *data logger*. Sehingga

dapat meminimalisir hal - hal yang tidak diinginkan seperti kelalaian membawa kunci sehingga harus merusak gembok panel dengan linggis atau benda tajam dan pencurian listrik maupun material listrik dengan membobol panel distribusi.

1.3 Batasan Masalah

Adapun batasan masalah dalam Tugas Akhir ini adalah :

1. Sistem penguncian diimplementasikan pada panel distribusi.
2. Pemrograman menggunakan *software* IDE untuk Arduino .
3. Komunikasi antara mikrokontroler dan *smartphone* Android menggunakan perantara *bluetooth* HC - 05 serta pemrograman aplikasi menggunakan MIT APP *Inventor* 2.
4. Delphi 7 digunakan sebagai *software* untuk menampilkan hasil *data logger*.
5. Sistem ini diujikan berjumlah 5 data ID dengan 2 ID terbaca benar, 2 ID terdaftar dan 1 ID Error.

1.4 Tujuan

Penelitian ini bertujuan untuk merancang dan membuat sistem *lock door* panel listrik menggunakan *fingerprint scanner* dengan tujuan khusus sebagai berikut :

1. Merancang dan membuat sistem pengunci panel dengan teknologi sensor sidik jari *(fingerprint scanner)*.
2. Monitoring *data logger* dengan mengetahui siapa saja dan kapan saja pintu panel terbuka.

1.5 Sistematika Laporan

Pembahasan Tugas Akhir ini dibagi menjadi lima Bab dengan sistematika sebagai berikut:

Bab I : Pendahuluan

Bab ini meliputi latar belakang, permasalahan, tujuan penelitian, metodologi penelitian, sistematika laporan, dan relevansi.

Bab II : Teori Dasar

Bab ini memberikan penjelasan singkat tentang tinjauan pustaka, sistem penguncian panel, sensor pemindai sidik jari, Arduino Mega 2560, Modul

Bluetooth HC - 05, *Solenoid doorlock*, dan *software* MIT App *Inventor* 2.

Bab III : Perancangan Sistem
Bab ini membahas desain *hardware* serta perancangan *flowchart* dan pemrograman *software* berdasarkan teori dasar pada Bab II.

Bab IV : Simulasi, Implementasi dan Analisis Sistem
Bab ini memuat hasil simulasi dan implementasi serta analisis dari hasil tersebut.

Bab V : Penutup
Bab ini berisi kesimpulan dan saran dari hasil pembahasan yang telah diperoleh.

1.6 Relevansi

Hasil yang diperoleh dari Tugas Akhir ini diharapkan menjadikan sebuah sistem *lock door* pintu panel yang efektif untuk menekan jumlah kerugian dari tindak kriminalitas dan pengamanan jalur distribusi di sekitar masyarakat.

--- Halaman ini sengaja dikosongkan ---

BAB II
TEORI DASAR

2.1 Tinjauan Pustaka[1][2][3]

Ada banyak metode yang pernah diajukan untuk menyelesaikan permasalahan yang timbul pada sistem keamanan pintu. Di antaranya pada Tugas Akhir Gayung yang berjudul "Sistem Pengaman Rumah dengan *Security Password* Menggunakan Sensor Gerak Berbasis Mikrokontroler AT89S51", digunakan *security password* sebagai kode yang harus dimasukkan. Hasil yang dicapai terdapat kekurangan, yaitu sistem *password* dianggap kurang praktis karena pengguna harus menghafal digit *password* yang ada.

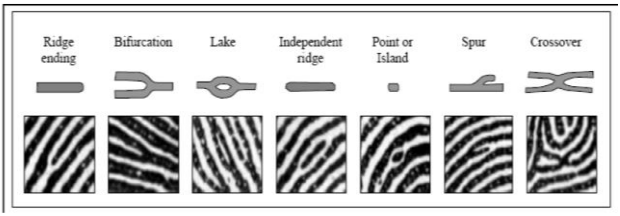
Pada skripsi Dwirani yang berjudul "Implementasi Sidik Jari untuk Sistem Absensi pada SDN II Cineam Tasikmalaya" digunakan *fingerprint scanner* sebagai sistem absensi yang ada pada perusahaan. Pada Tugas Akhir ini dilakukan perancangan sistem *lock door* pintu panel menggunakan pemindai sidik jari berbasis Android. Hal ini didasarkan pada skripsi Suroto yang berjudul "Studi Penyempurnaan Identifikasi Sidik Jari pada *Algoritma Minutea".* Sistem pemindai sidik jari dianggap sangat sensitif, sehingga setiap orang di dunia tidak akan memiliki kode sidik jari yang sama. Sidik jari dianggap sebagai bentuk identitas yang khusus dan spesifik, sehingga kemungkinan terjadinya sabotase pada suatu sistem adalah sangat kecil.

2.2 *Fingerprint Scanner*[4]

Fingerprint scanner atau alat pemindai sidik jari merupakan suatu komponen elektronik yang berfungsi membaca pola sidik jari dan mencocokkan dengan *database* yang telah tersimpan dalam memori. Sistem ini memiliki cara kerja yang terbagi menjadi tiga yaitu *enrollment, delete,* dan *fingerprint acces*s. Pada saat ini, sensor sidik jari telah banyak dipasarkan. Divisi forensik pada penegak hukum saat ini menggunakan sidik jari sebagai proses penyelidikan.

Hal ini dikarenakan setiap orang memiliki garis atau bentuk sidik jari yang berbeda – beda. Pola pembacaan sidik jari dapat dibandingkan dengan metode *minutiea*. Metode ini sering digunakan sebagai metode perbandingan sensor *fingerprint* berdasarkan pencocokan pola *minutiea*. *Minutiea* yang berarti kecil, dalam konteks sidik jari hal tersebut berarti pola – pola pada sidik jari. Sebagian besar metode ini memerlukan

gambar sidik jari untuk dikonversi menjadi gambar biner. **Gambar 2.1** merupakan contoh pola metode *minutiea*.



Gambar 2.1 Tujuh Jenis Umum Pola pada Metode *Minutiea*

Pada saat sistem memulai *scanning* sidik jari dengan alat pemindai, hasil akan disimpan dalam memori. Hasil *scanning* disimpan dalam format digital pada saat *enrollment* (pendaftaran sidik jari). Setelah itu, rekaman sidik jari tersebut diproses dan dibuatkan daftar pola fitur sidik jari yang unik. Pola fitur sidik jari yang unik tersebut kemudian disimpan dalam memori atau *database*. Pola sidik jari yang unik ini disebut dengan pola *minutiea*. Pada saat identifikasi, pola *minutiea* tersebut kemudian dicocokkan dengan hasil *scan* sidik jari.

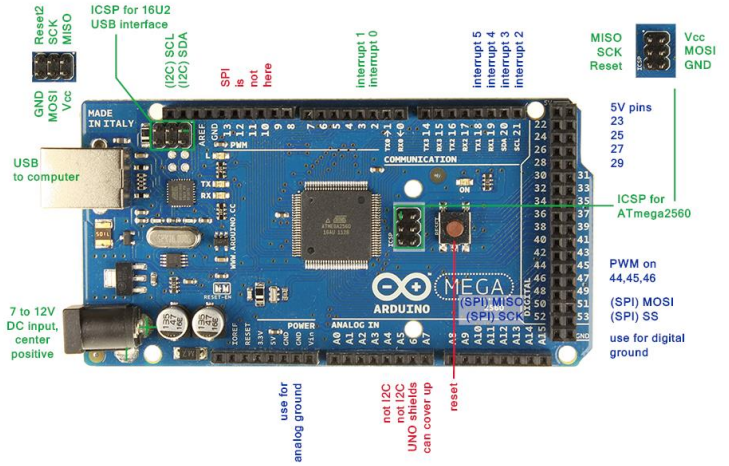
Teknik pembacaan sidik jari oleh mesin absensi sidik jari dapat dikelompokkan menjadi empat tipe, yaitu teknik optik, ultrasonik, kapasitansi dan *thermal*. Kelemahan metode ini adalah hasil *scanning* sangat tergantung dari kualitas sidik jari. Jika kualitas sidik jari luka, kering, dan tidak jelas, maka kualitas hasil pembacaan tidak bagus. Kelemahan lain adalah teknik ini bisa diakali dengan jari palsu. Tapi teknik ini mempunyai keuntungan mudah dilakukan dan tidak membutuhkan biaya yang mahal.

Gambar 2.2 *Fingerprint Scanner* dan *Connector*

Pada **Gambar 2.2** di atas merupakan contoh modul pemindai sidik jari yang dapat dikoneksikan dengan Arduino. Dengan DSP prosesor yang berkecepatan tinggi, modul *fingerprint scanner* ini juga dapat diaplikasikan dengan serial *device* yang lain, seperti MSP430, AVR, PIC, STM32, ARM dan FPGA *device*. Modul ini memiliki memori yang dapat menyimpan hingga 1000 data sidik jari. *Fingerprint scanner* memiliki kemampuan pembacaan sidik jari dengan tingkat sensitivitas yang tinggi baik dalam keadaan basah. Selain itu alat ini memiliki kecepatan tinggi saat melakukan sistem pemindaian, pencarian dan pembandingan pola sidik jari.

Dengan fitur - fitur dan segala keunggulan tersebut, *fingerprint scanner* ini dapat difungsikan dalam berbagai bidang, terutama yang bersangkutan dengan masalah keamanan. Dapat difungsikan sebagai piranti absensi modern, saklar elektronik maupun pengganti *password* dengan sensitivitas yang tinggi.

2.3 Arduino MEGA 2560



Gambar 2.3 Arduino MEGA 2560

Gambar 2.3 seperti di atas merupakan contoh *board* Arduino MEGA. Arduino Mega 2560 adalah *board* berbasis mikrokontroler pada ATMega 2560. Board ini memiliki pin I/O sejumlah 54 buah digital I/O pin (15 pin diantaranya adalah PWM), 16 pin analog input, 4 pin UART

(serial *port hardware*). Arduino Mega 2560 dilengkapi dengan sebuah *oscillator* 16 Mhz, sebuah *port* USB, *power jack* DC, ICSP *header*, dan tombol *reset*. Spesifikasi tersebut dapat dilihat pada **Tabel 2.1.**

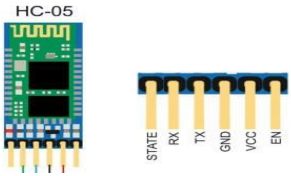
Pin – pin ini berisi semua yang diperlukan untuk mendukung mikrokontroler, hanya terhubung ke komputer dengan kabel USB atau sumber tekanan bisa didapat dari adaptor AC – DC atau baterai untuk menggunakannya (Arduino, Inc., 2009).

Tabel 2.1 Deskripsi Spesifikasi pada Arduino MEGA

Mikrokontroler	**ATMega 2560**
Operating Voltage	**5 V**
Input Voltage (recommended)	**7 – 12 V**
Input Voltage (limit)	**6 – 20 V**
Digital I/O Pin	**54 (14 diantaranya *input* PWM)**
Analog *Input* Pin	**16**
DC *Current* per I/O Pin	**40 mA**
DC *Current for* 3,3V Pin	**50 mA**
Flash Memory	**256 KB, 8 KB *bootloader***
RAM	**8 KB**
EEPROM	**4 KB**
Clock Speed	**16 MHz**

2.4 Modul *Bluetooth* HC – 05

Bluetooth Module HC – 05 merupakan *module* komunikasi nirkabel pada frekuensi 2.4GHz dengan pilihan koneksi sebagai *slave* ataupun sebagai *master*. *Master* berarti modul yang menginisiasi koneksi, sedangkan *slave* berarti menerima inisiasi. Konfigurasi dapat dilihat pada **Gambar 2.4.**



Gambar 2.4 Konfigurasi Pin Modul *Bluetooth* HC-05

Untuk dapat bertukar data, maka harus melakukan *pairing*. *Pairing* merupakan proses pencarian perangkat oleh *discover* (pencari) pada *discoverable* (yang dicari), serta melakukan autentikasi untuk mengenali perangkat lain. *Interface* yang digunakan adalah serial RXD, TXD, VCC, dan GND. Tegangan *input* antara 3.6 ~ 6V, arus saat *unpaired* ±30mA, dan saat *paired* (terhubung) ±10mA. Pin *interface* 3.3V dapat langsung dihubungkan ke berbagai macam mikrokontroler (khusus Arduino, 8051, 8535, AVR, PIC, ARM, MSP430, etc.).

2.5 *Solenoid Doorlock*

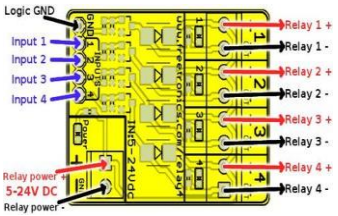
Solenoid doorlock adalah alat elektronik yang dibuat khusus untuk pengunci pintu. Alat ini sering digunakan pada kunci pintu otomatis. *Solenoid* ini akan bekerja apabila diberi tegangan. Tegangan *solenoid doorlock* ini rata-rata yang dijual dipasaran adalah 12 volt tapi ada juga yang 6 volt dan 24 volt.

Gambar 2.5 *Solenoid Doorlock*

Pada Tugas Akhir ini menggunakan *solenoid* 12V dengan ukuran seperti pada **Gambar 2.5.** Pada kondisi normal *solenoid* dalam posisi tuas terkunci. Jika diberi tegangan *solenoid* dalam posisi tuas terbuka. Cocok dipakai untuk pengunci pintu, lemari maupun brankas.

2.6 *Relay*

Relay adalah sakelar yang memanfaatkan tarikan medan magnet yang dibentuk oleh *coil* dari *relay* itu sendiri. Di dalam *relay* terdapat beberapa sakelar atau *switch* NO (*Normally Open*) atau NC (*Normally Close*) yang akan berubah keadaan setelah *coil* mendapat tegangan dari luar. Selama *coil* diberi tegangan selama itu pula perubahan keadaan *switch*. Pada sistem ini menggunakan *relay* empat modul dengan konfigurasi relay seperti pada **Gambar 2.6.**

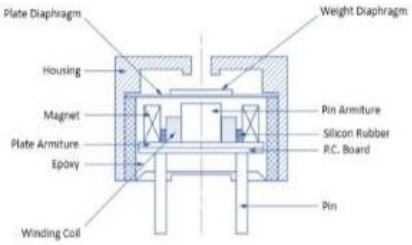


Gambar 2.6 Konfigurasi *Relay* Empat Modul

Didalam *relay* terdapat *cover* sebagai pelindung, dan terminal sebagai sambungan *coil* maupun *switch*. Di sekitar terminal pun ada nomor kaki yang sesuai pada skema *relay* pada *cover*.

2.7 *Buzzer*

Buzzer merupakan komponen elektronika yang berfungsi untuk mengubah getaran listrik menjadi getaran suara. Prinsip kerja *buzzer* seperti pada **Gambar 2.7** yakni terdiri dari kumparan yang terpasang pada diafragma dan kemudian kumparan tersebut dialiri arus sehingga menjadi elektromagnet. Kumparan tersebut akan tertarik ke dalam atau keluar, tergantung dari arah arus dan polaritas magnet.



Gambar 2.7 Struktur *Buzzer*

Karena kumparan dipasang pada diafragma maka setiap gerakan kumparan akan menggerakkan diafragma secara bolak - balik sehingga membuat udara bergetar yang akan menghasilkan suara. *Buzzer* biasa digunakan sebagai indikator bahwa proses telah selesai atau terjadi suatu kesalahan pada sebuah alat (*alarm*).

2.8 EEPROM

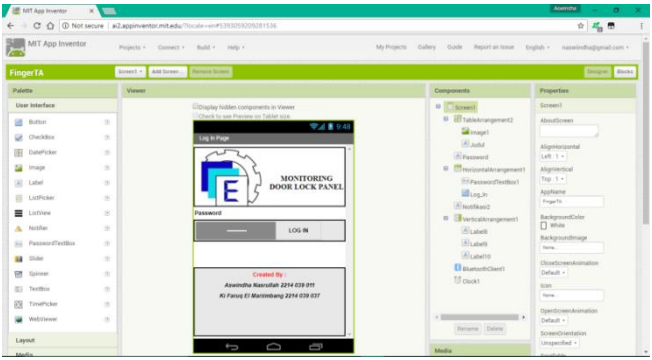
Serial EEPROM tipe 24xx adalah merupakan memori serial yang menggunakan teknologi I2C di mana dengan adanya penggunaan teknologi tersebut, jumlah I/O yang digunakan untuk mengakses memori tersebut semakin sedikit. Hal ini sangat bermanfaat bagi sebuah sistem yang memerlukan banyak I/O.

Gambar 2.8 Modul EEPROM AT24C02

Penggunaan I/O yang semakin sedikit untuk mengakses memori, akan menyediakan lebih banyak I/O yang dapat digunakan untuk keperluan lain. I2C adalah teknologi komunikasi serial yang ditemukan oleh Philips pada tahun 1992 dan direvisi hingga versi 2.1 yang terbaru pada tahun 2000. Teknologi ini hanya menggunakan 2 buah jalur I/O yaitu SDA dan SCL. Pada Tugas Akhir ini menggunakan modul EEPROM AT24C02 seperti pada **Gambar 2.8.**

2.9 MIT APP *Inventor* 2

MIT APP *Inventor* adalah aplikasi *web* sumber terbuka yang awalnya dikembangkan oleh Google, dan saat ini dikelola oleh Massachusetts Institute of Technology (MIT). Pada **Gambar 2.9** merupakan tampilan awal pada MIT APP *Inventor* 2. Dalam menciptakan APP *Inventor*, Google telah melakukan riset yang berhubungan dengan komputasi edukasional dan menyelesaikan lingkungan pengembangan online Google.



Gambar 2.9 Tampak Awal pada MIT APP *Inventor* 2

Aplikasi ini memungkinkan pengguna baru untuk memprogram komputer dan aplikasi perangkat lunak yang digunakan sebagai sistem operasi Android. Pengguna dapat menggunakan tampilan grafis GUI dan fitur *drag* dan *drop* visual objek untuk membuat sebuah aplikasi dapat berjalan pada sistem operasi Android.

MIT APP *Inventor* 2 pada dasarnya bekerja secara *online* melalui *browser* internet, tetapi bisa dilakukan dengan cara *offline*, dengan beberapa cara yang agak sulit. Halaman kerja MIT APP *Inventor* 2 memiliki 2 bagian besar yaitu :

1. Komponen Desainer

Penggunaan komponen desainer seperti pada **Gambar 2.10** digunakan untuk memilih komponen guna membangun aplikasi. Komponen ini terbagi menjadi empat jenis komponen yaitu *palette*, *viewer*, *components*, *properties*, *main menu*, dan *media.*

Gambar 2.10 Tampilan pada Komponen Desainer

2. *Block Editor*

Block editor seperti pada **Gambar 2.11.**merupakan halaman kerja utama dari MIT APP *Inventor* 2. *Block editor* terdiri atas beberapa bagian, yaitu *main menu*, *blocks*, dan *viewer*.

Gambar 2.11 Tampilan pada Komponen *Block*

-----Halaman ini sengaja dikosongkan-----

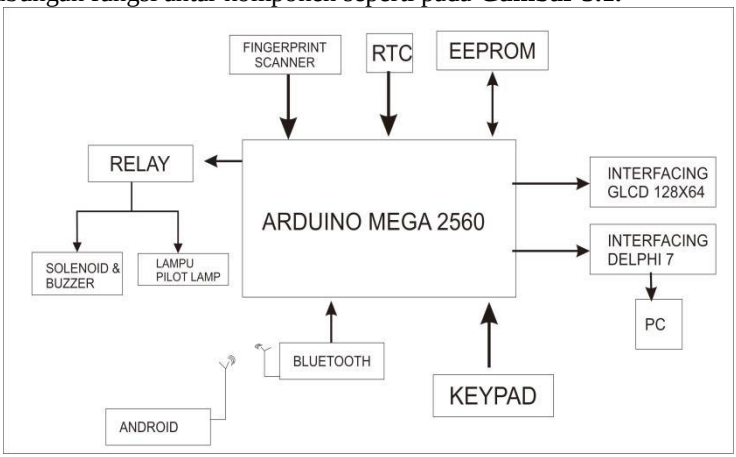
BAB III
PERANCANGAN DAN PEMBUATAN ALAT

Pada bab ini membahas tentang tahapan perancangan dan pembuatan alat yang dilakukan terhadap Tugas Akhir yang berjudul "Sistem *Door Lock* Panel Listrik Menggunakan *Fingerprint Scanner* berbasis Android". Perancangan dan pembuatan sistem meliputi perangkat keras dan perangkat lunak. Perangkat keras meliputi perancangan *box* panel dan *solenoid* serta *keypad*, perancangan *wiring* pada sensor *fingerprint* , perancangan *relay* sebagai *driver solenoid* dan *buzzer*, perancangan *wiring* GLCD, perancangan RTC (*Real Time Clock*) DS1307, dan pembuatan rangkaian seluruh komponen pada *Shield* Arduino Mega 2560.

Perangkat lunak meliputi program *fingerprint*, kontrol aktuator (*solenoid)* dan *buzzer*, tampilan GLCD, RTC DS1307, EEPROM sebagai memori penyimpanan *data logger*, kontrol *keypad*, dan komunikasi *bluetooth* HC – 05.

3.1 Blok Fungsional Sistem

Berdasarkan tinjauan pustaka dan dasar teori pada bab II, maka dapat disimpulkan perancangan dan pembuatan sistem yang digunakan dengan menggunakan sebuah blok fungsional sistem yang menjelaskan hubungan fungsi antar komponen seperti pada **Gambar 3.1.**



Gambar 3.1 Diagram Fungsional pada Sistem

Berikut penjelasan mengenai blok diagram pada **Gambar 3.1** yaitu:

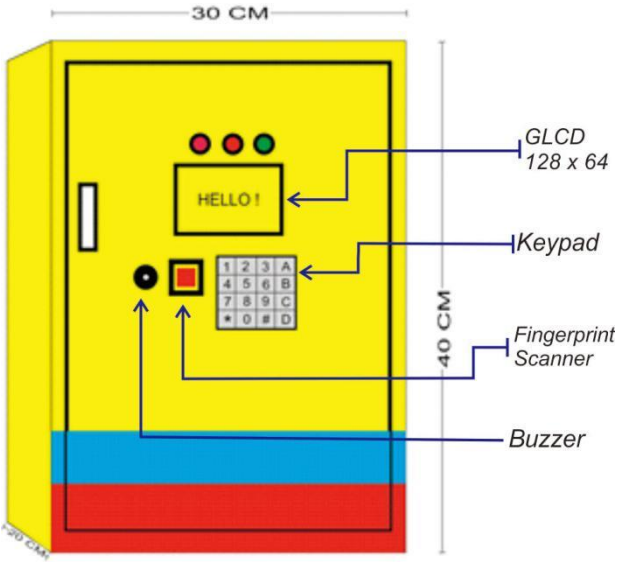
1. Pemindai sidik jari (*fingerprint scanner*), digunakan untuk mengetahui kecocokan sidik jari pengguna dengan *database* yang ada di memori Arduino Mega 2560, merupakan mikrokontroler yang berfungsi sebagai pengontrol perangkat elektronik dan dapat menyimpan program didalamnya.
2. *Solenoid doorlock* digunakan sebagai aktuator untuk menjaga pintu tetap terkunci atau terbuka.
3. *Bluetooth HC - 05* digunakan sebagai media komunikasi untuk mengirim perintah dari *smartphone* ke Arduino.
4. Android sebagai jenis *smartphone* yang akan mengatur kerja awal sistem dan pembuatan aplikasi menggunakan MIT APP *Inventor* 2.
5. *Keypad* digunakan sebagai *input* digit angka pada pintu panel.
6. Delphi 7 digunakan sebagai program menampilkan hasil *database.*
7. EEPROM merupakan media penyimpanan data yang akan menyimpan data berupa ID, status sistem, jam dan tanggal.
8. RTC digunakan untuk mengatur waktu dan tanggal.
9. Pilot *lamp* sebagai indikator adanya sumber tegangan dan indikator saat *log in* sukses berupa cahaya lampu.
10. *Buzzer* digunakan sebagai indikator untuk menandakan sukses akses *solenoid* berupa suara.
11. Tampilan GLCD berupa menu pilihan yang nantinya dapat membantu proses yang diinginkan mengunakan bantuan *keypad* 4x4.
12. *Data logger* ditampilkan berupa ID, status sistem, serta waktu yang menggunakan modul RTC. Hasil tersebut ditampilkan pada PC untuk mengetahui siapa dan kapan seseorang masuk ruangan.

3.2 Perancangan Perangkat Keras

Pada perancangan perangkat keras ini, prosesnya dibagi menjadi dua bagian, yaitu perancangan mekanik dan elektrik. Masing-masing perancangan tersebut selanjutnya akan dibahas lebih mendalam pada sub bab berikutnya.

3.2.1 Perancangan Mekanik

Perangkat untuk membuka pintu dengan *fingerprint scanner* tersebut merupakan alat untuk membuka kunci pintu *solenoid doorlock* dengan kemudahan dan tingkat kepraktisan yang tinggi. Secara umum perangkat keras ini terdiri dari *fingerprint scanner* sebagai masukan, penampil data berupa *Graphic* LCD, sistem mikrokontroler, pintu yang sudah didesain sedemikian rupa seperti **Gambar 3.2** di bawah ini.



Gambar 3.2 Desain Sistem pada *Box* Panel

Untuk menyesuaikan desain alat agar dapat bekerja maksimal, maka diperlukan desain kunci dan *handle* pintu yang tepat. Komponen *solenoid doorlock* harus menempel di belakang pintu agar alat dapat bekerja sesuai fungsinya.

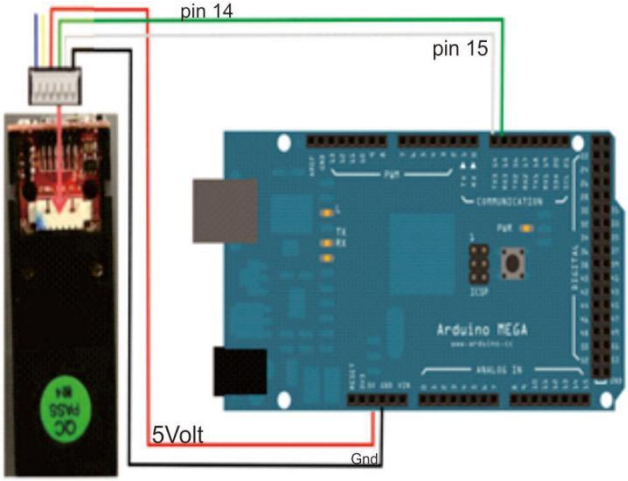
Pemasangan *fingerprint scanner* akan diletakkan di sisi depan pintu. Dilengkapi dengan *buzzer* yang akan menjadi indikator kecocokan sidik jari pengguna dengan *data logger* yang telah tersimpan dalam memori sensor, serta tampilan GLCD untuk memilih menu eksekusi sistem.

3.2.2 Perancangan Elektronik

Perancangan elektronik ini meliputi desain PCB serta pengkabelan. Pada perancangan elektronik dibagi menjadi beberapa sub bab yang akan dijelaskan per sub bab nya, antara lain :

3.2.2.1 Perancangan *wiring Fingerprint*

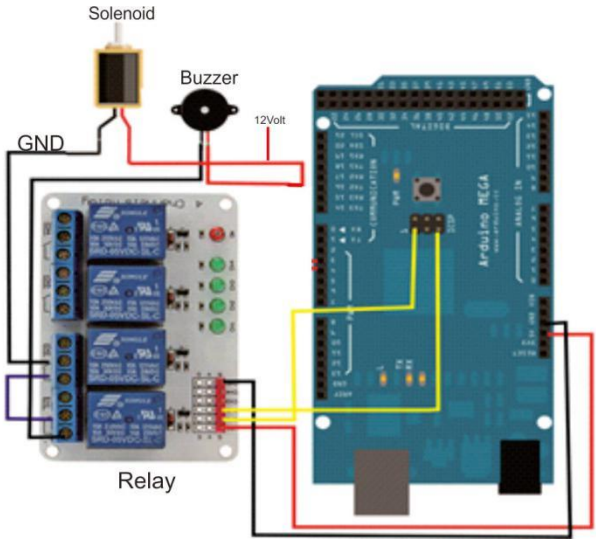
Pada **Gambar 3.3** menunjukkan koneksi dari *fingerprint scanner* ke Arduino Mega 2560. Pin 1 dengan kabel warna hitam adalah *ground*. Pin 2 dan pin 3 dengan warna putih dan hijau adalah data *input* dan data *output*. Dikoneksikan dengan pin 14 dan pin 15 dari Arduino Mega 2560. Pin 4 dengan kabel warna merah untuk *input* 5 Volt. Sedangkan pin 5 dan pin 6 adalah untuk *induction signal output* dan *touch induction power input* (biasanya diabaikan).



Gambar 3.3 Koneksi *Fingerprint Scanner* ke Arduino

3.2.2.2 Perancangan *Relay* sebagai Driver *Solenoid* dan *Buzzer*

Solenoid merupakan aktuator yang memiliki karakteristik kerja pada tegangan 12 volt dengan besar arus tertentu. Untuk menguatkan arus yang diterima oleh *solenoid* dari arduino, maka diperlukan sebuah *driver*. Untuk itu, pada **Gambar 3.4** koneksi *solenoid* dan *buzzer* menggunakan *relay* sebagai *driver* penguat, *switch* dan pengaman tegangan dan arus.



Gambar 3.4 Koneksi *Solenoid*, *Buzzer*, dan *Relay* ke Arduino

3.2.2.3 Perancangan Rangkaian *Shield* Arduino MEGA 2560

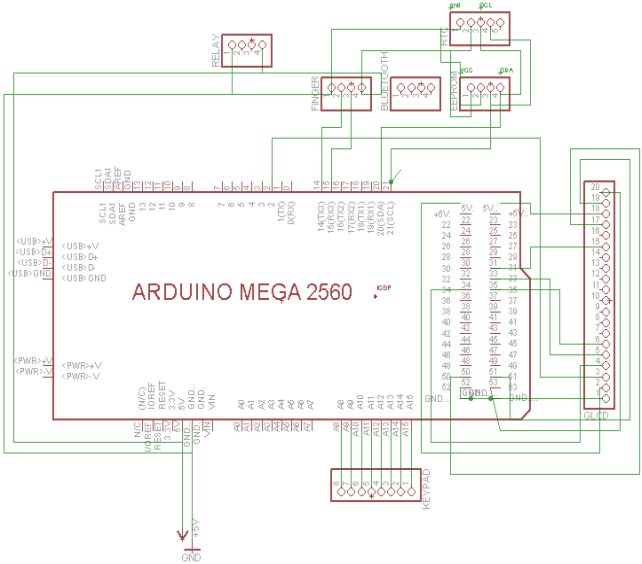
Pada perancangan rangkaian ini menggunakan Arduino MEGA 2560 untuk menerima masukan dari sensor *fingerprint scanner*, GLCD 128 x 64 untuk menampilkan hasil, menerima data waktu dari rangkaian RTC, menulis dan membaca data EEPROM, serta menjalankan program komunikasi antara sistem dan Android menggunakan *bluetooth HC - 05*. Arduino Mega sebagai mikrokontroler diprogram untuk mengontrol *solenoid* yang berada di pintu box panel, dan menyimpan data dalam EEPROM. Jumlah pin yang digunakan pada sistem ini seperti pada **Tabel 3.1.**

Tabel 3.1 Mapping Pin Analog/Digital yang Digunakan

No	Modul	PIN ANALOG/DIGITAL
1.	RTC	VCC : 5V GND SDA : DIGITAL 20 SCL : DIGITAL 21

No	Modul	PIN ANALOG/DIGITAL
2.	EEPROM	VCC : 5V GND SDA : DIGITAL 20 SCL : DIGITAL 21
3.	*BLUETOOTH*	VCC : 5V GND RX : DIGITAL 16 (TX2) TX : DIGITAL 17 (RX2)
4.	*GRAPHIC LIQUID CRYSTAL DISPLAY* 128X64	VSS : GND VDD : VCC 5V VO: DIGITAL 2 RS: DIGITAL 36 RW: DIGITAL 35 E: DIGITAL 37 PSB : DIGITAL 33 RST : DIGITAL 52 VOUT : VCC 5V BLA : DIGITAL 53 BLK : GND
5.	*RELAY*	GND :3 ICSP IN1 : DIGITAL 3 IN2 : DIGITAL 4 VCC :1 ICSP
6.	*KEYPAD*	PIN : A8 - A15

Berdasarkan **Tabel 3.1** maka diperlukan sebuah rangkaian berbentuk *shield* yang akan memudahkan dalam pengaturan sensor dan komponen pendukung. Rangkaian pada sistem ini seperti pada **Gambar 3.5** untuk mempermudah koneksi antar komponen sistem dengan Arduino. Selain itu dengan adanya modul ini meminimalisir jumlah *wiring* pada sistem ini.



Gambar 3.5 Rangkaian Modul *Shield* pada Arduino Mega 2560

3.3 Perancangan *Software*

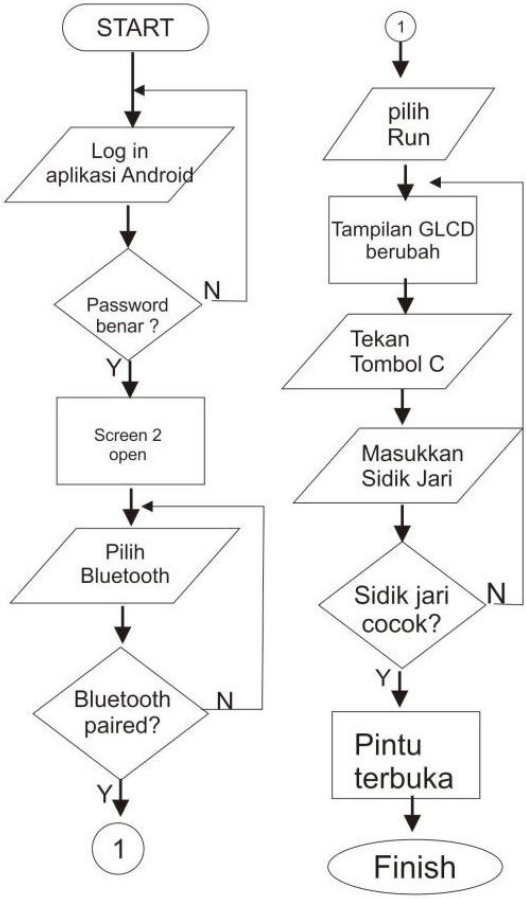
Perancangan *software* yang digunakan yaitu pada program Arduino Mega 2560 untuk membuat dan merencanakan program dalam bahasa C menggunakan IDE Arduino, program Android menggunakan *block* diagram pada MIT APP *Inventor* 2, dan program Delphi 7 untuk menampilkan hasil EEPROM ke PC.

Dalam perancangan *software* Arduino dengan fungsi terkait diperlukan beberapa tahapan yang harus dilakukan terlebih dahulu. Dengan membuat algoritma setelah membuat *flowchart* sistem ini agar program lebih sederhana. Setelah itu barulah memprogram menggunakan bahasa C.

3.4 Arduino IDE

Pada **Gambar 3.6** menjelaskan alur program sistem. Pada posisi awal sistem diperlukan *log in* melalui Android untuk masuk pada sistem utama *fingerprint*. Jika *password* pada aplikasi Android menyatakan benar, maka membuka *screen* 2 pada aplikasi tersebut untuk menghubungkan komunikasi *bluetooth* antara *smartphone* dan Arduino

Mega 2560. Pada aplikasi tombol *RUN* akan mengaktifkan sistem utama akses *fingerprint* dengan menampilkan pilihan menu A, B, C, dan D.



Gambar 3.6 *Flowchart* pada Sistem *Lock Door*

Pada saat menekan tombol C pada *keypad*, maka akan dilakukan pencocokan sidik jari oleh *fingerprint*. Jika sidik jari sesuai dengan *database* sensor , maka *solenoid* akan aktif dan pintu terbuka, jika tidak maka akan kembali ke pilihan menu A, B, C, dan D.

3.4.1 *Fingerprint Access*

Pada **Gambar 3.7** merupakan inisialisasi pengunaan *library* dan pin serial pada Arduino Mega 2560. Berdasarkan gambar tersebut maka dapat diketahui bahwa pin untuk sensor *fingerprint* menggunakan komunikasi serial melalui pin 15 sebagai RX yang disambungkan pada kabel putih sensor (TD) dan pin 14 sebagai TX yang disambungkan pada kabel hijau sensor (RD).

fingerprint_TA

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>

int getFingerprintIDez();

// pin #2 is IN from sensor (GREEN wire)
// pin #3 is OUT from arduino  (WHITE wire)
//SoftwareSerial mySerial(2, 3);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Seria

// On Leonardo/Micro or others with hardware serial, use
//Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Ser
```

Gambar 3.7 Inisialisasi *Fingerprint Scanner*

3.4.2 *Enrollment*

Sistem ini didesain agar hanya bisa diakses oleh orang yang telah didaftarkan oleh *admin*. Untuk melakukan pendaftaran *user* baru, *admin* harus menjalankan program *enrollment* sidik jari menggunakan program seperti pada **Gambar 3.8.**

```
  while (!  getFingerprintEnroll() );
}

uint8_t getFingerprintEnroll() {

  int p = -1;
  Serial.print("Waiting for valid finger to enroll as #"); Serial.pr
  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image taken");
      break;
    case FINGERPRINT_NOFINGER:
      Serial.println(".");
      break;
```

Gambar 3.8 Program *Enrollment* Sidik Jari

3.4.3 *Delete*

Pada **Gambar 3.9** merupakan program untuk penghapusan sidik jari untuk *user* yang telah tidak terpakai. Pada saat penghapusan ID, *admin* akan memasukkan nomer ID yang akan dihapus. Setelah *user* ID dihapus dari memori, maka *user* tersebut tidak dapat mengakses sistem keamanan menggunakan *fingerprint scanner* ini.

```
  deleteFingerprint(id);
}

uint8_t deleteFingerprint(uint8_t id) {
  uint8_t p = -1;

  p = finger.deleteModel(id);

  if (p == FINGERPRINT_OK) {
    Serial.println("Deleted!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_BADLOCATION) {
    Serial.println("Could not delete in that locati
    return p;
```

Gambar 3.9 Program Penghapusan ID *Fingerprint Scanner*

3.5 MIT APP *Inventor* 2

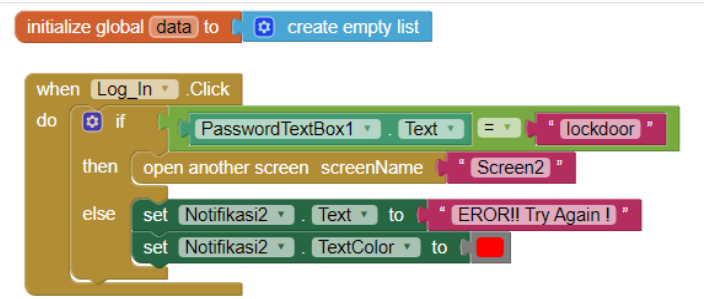
Aplikasi ini didesain agar tidak semua orang bisa mengakses sistem *fingerprint scanner* dan melalui Android untuk mengaktifkan sistem pada pintu panel. Hanya pengguna Android yang telah memiliki aplikasi ini dan mengetahui *password* aplikasi untuk mengaktifkan sistem. Pada aplikasi sistem ini menggunakan dua *screen*. Dengan deskripsi yaitu pada *screen* pertama sebagai tampilan awal aplikasi dan *screen* kedua sebagai tampilan utama pada aplikasi sistem ini.

3.5.1. *Screen Pertama (Log In Page)*

Pada **Gambar 3.10** dapat dilihat bahwa aplikasi ini memerlukan *log in* menggunakan *password* yang hanya diketahui oleh *admin* atau pemilik aplikasi. Jika *password* benar, maka menampilkan *screen* selanjutnya. Namun jika *password* salah, maka menampilkan tulisan "*ERORI! Try Again !".*

Gambar 3.10 Tampilan *Screen* Pertama pada Android

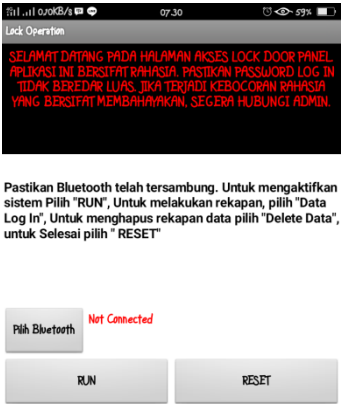
Untuk menghubungkan *screen* pertama dan *screen* kedua diperlukan pemrograman seperti pada **Gambar 3.11**. Jika *password text box* mendeteksi masukkan "*lockdoor"* maka *screen* pertama akan berganti menjadi *screen* 2. Jika *password text box* mendeteksi masukkan selain "*lockdoor"* maka tampilan akan tetap pada *screen* pertama dan memunculkan *notification "EROR!!Try Again!".*



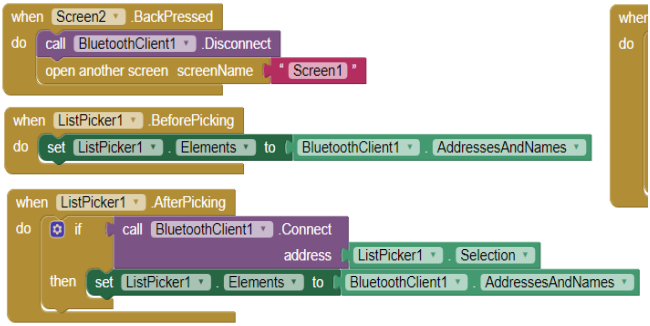
Gambar 3.11 Pemrograman MIT APP *Inventor* 2 pada *Screen 1*

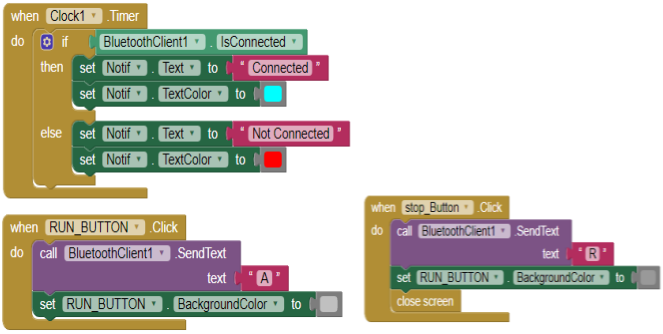
3.5.2. *Screen Kedua (Log In Page)*

Pada **Gambar 3.12** dapat dilihat bahwa pada *screen* ini menampilkan tombol "Pilih *Bluetooth"*, tombol "*RUN*" dan tombol "*RESET*" yang masing - masing memiliki fungsi berbeda – beda. Tombol "Pilih *Bluetooth*" yang digunakan untuk memilih *bluetooth*, tombol "*RUN*" yang digunakan untuk mengaktifkan sistem utama *fingerprint* sensor pada pintu panel, dan tombol "*RESET*" yang digunakan untuk menonaktifkan sistem utama dengan menampilkan kembali tampilan awal sistem "SELAMAT DATANG" pada GLCD. Pemrograman blok diagram *screen* kedua seperti pada **Gambar 3.13.**



Gambar 3.12 Tampilan *Screen 2* pada Android



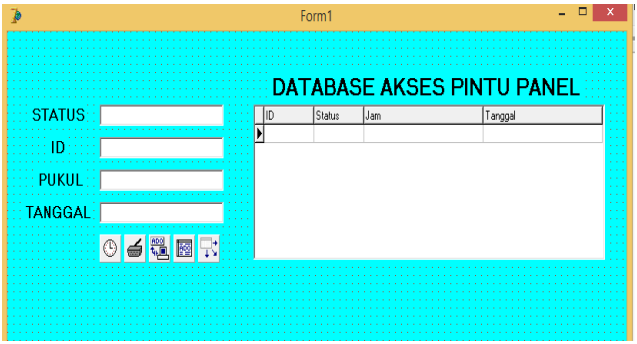


Gambar 3.13 Pemrograman MIT APP *Inventor* 2 pada *Screen 2*

3.6 Delphi 7

Database adalah kumpulan data yang disimpan secara sistematis di dalam PC dan dapat diolah menggunakan program aplikasi untuk menampilkan informasi sebuah penyimpanan data. *Database* sistem ini mencakup riwayat *log in* sistem *lock door* yang kemudian ditampilkan pada PC menggunakan aplikasi Delphi 7.

Tampilan aplikasi Delphi 7 pada *database* sistem ini seperti pada **Gambar 3.14** dengan format kolom tabel yaitu "ID" sebagai pengenal sidik jari yang telah didaftarkan dan disimpan pada memori sensor *fingerprint*, "Status" digunakan sebagai informasi berhasil atau tidak berhasil saat akses pintu. Jika berhasil maka akan menampilkan "1", jika tidak berhasil maka akan menampilkan "0", kolom "Jam" dan "tanggal" digunakan sebagai informasi kapan akses pintu dilakukan.



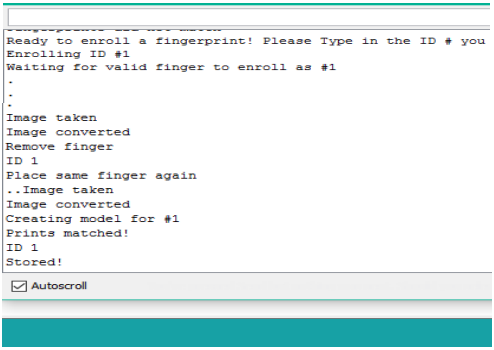
Gambar 3.14 Tampilan Aplikasi pada Delphi 7

-----Halaman ini sengaja dikosongkan-----

BAB IV
PENGUJIAN DAN ANALISA DATA

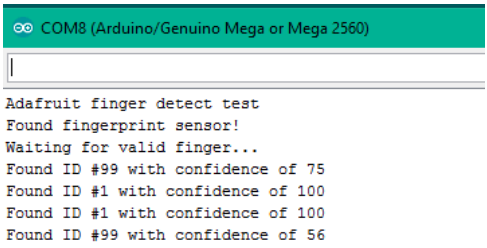
4.1 Pengujian *Fingerprint Scanner*

Fingerprint scanner diuji dengan menggunakan program yang diupload ke dalam Arduino. Ada beberapa macam program yang digunakan untuk menguji *fingerprint scanner*, diantaranya pendaftaran sidik jari (*enrollment*) dan pengujian kecocokan sidik jari.



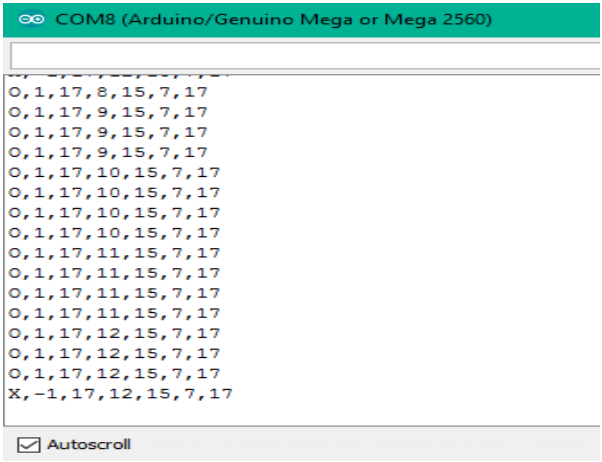
Gambar 4.1 Pengujian *Enrollment* pada *Fingerprint*

Pada pengujian **Gambar 4.1** menunjukkan bahwa saat sidik jari telah berhasil mendapatkan *image* sidik jari maka sidik jari tersebut telah terdaftar dan tersimpan sesuai dengan ID yang dimasukkan. Untuk mengetahui keakuratan pembacaan sensor *fingerprint* maka diperlukan pengujian *log in* dengan mencantumkan nilai *confidence* untuk mengetahui seberapa besar keakuratan posisi sidik jari.



Gambar 4.2 Pengujian Kecocokan Sidik Jari

Hasil pengujian pada **Gambar 4.2** menunjukkan kecocokan sidik jari dengan *fingerprint scanner*. Semakin tinggi nilai *confidence* pada serial monitor, maka semakin tinggi pula tingkat kecocokan sidik jari tersebut. Tingkat kecocokan dapat ditentukan oleh posisi sidik jari, kondisi sidik jari dan lain sebagainya, tergantung jenis *fingerprint scanner* yang digunakan.



Gambar 4.3 Pengujian *Error* pada Sensor *Fingerprint*

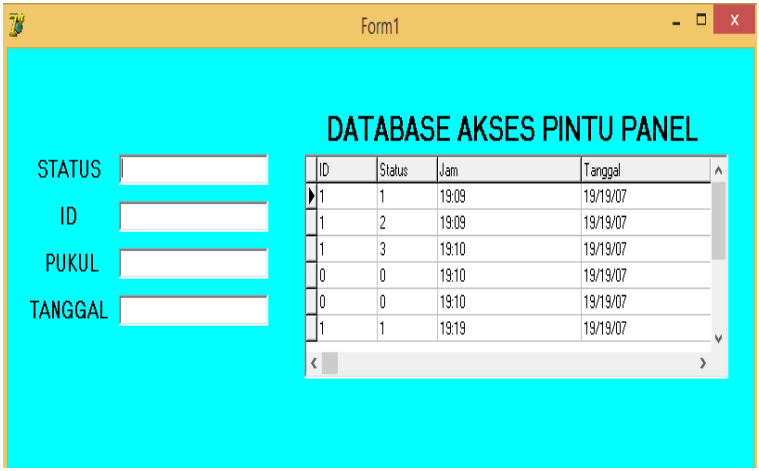
Pengujian *error* sidik jari dilakukan dengan melakukan *log in* pada ID yang sama secara berturut – turut. Hasilnya adalah seperti pada **Gambar 4.3** dengan format sebagai berikut:

O : Status Sukses
1 : ID
17 : *Hour*
9 : *Minute*
15 : *Day*
7 : *Month*
17 : Year
X : Status Gagal

Berdasarkan pengujian tersebut terlihat bahwa *log in* ke - 16 terjadi kegagalan. Hal ini dikarenakan oleh posisi jari pada saat melakukan *log in* kurang lembab ataupun tidak sesuai dengan posisi saat didaftarkan.

4.2 Pengujian Aplikasi Delphi 7

Delphi 7 diuji dengan menampilkan hasil penyimpanan *database* pada EEPROM dengan format ID, Status, Jam, dan Tanggal. Pengujian ini melalui PC yang telah memiliki aplikasi *database*..



Gambar 4.4 Pengujian Hasil Pembacaan *Database*

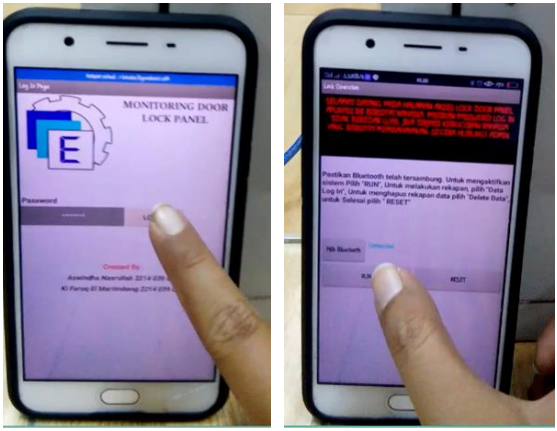
Berdasarkan pengujian seperti pada **Gambar 4.4** dapat terlihat bahwa *database* berhasil ditampilkan. Sebagai contoh dapat dilihat pada baris pertama yaitu pembacaan 1 sebagai ID dan saat berhasil melakukan *log in*, maka kolom Status tampil keterangan "1", kolom Jam dan Tanggal menunjukkan Pukul 19:09 tanggal 19 juli 2017. Dan pada baris ke - 4 pembacaan 0 sebagai ID, kolom Status menunjukkan keterangan gagal "0" pada pukul 19:10 tanggal 19 juli 2017.

4.3 Pengujian Sistem Keseluruhan

Pengujian keseluruhan pada sistem dilakukan setelah menggabungkan keseluruhan komponen. Dengan menguji kerja *fingerprint*, *buzzer*, *solenoid*, tampilan GLCD, Tampilan Delphi 7 dan lampu indikator. Sistem keseluruhan seperti pada **Gambar 4.5**.

Gambar 4.5 Tampilan Keseluruhan Sistem

Pada panel, terdapat GLCD yang difungsikan sebagai penampil pada pintu panel. Pada awal sistem menerima tegangan, GLCD akan menampilkan "SELAMAT DATANG" dengan mencantumkan jam dan pukul yang diletakkan dibawah tulisan. Setelah sistem diaktifkan melalui Android seperti pada **Gambar 4.6**, maka akan menampilkan tampilan GLCD selanjutnya.



Gambar 4.6 Tampilan Aplikasi pada Android

Seperti pada **Gambar 4.7** menunjukkan tampilan awal GLCD 128 x 64 pada saat *stand by* atau pada saat menerima tegangan dan sistem siap digunakan.

Gambar 4.7 Tampilan pada Awal GLCD

Tampilan GLCD selanjutnya akan memudahkan user dalam pemilihan eksekusi sistem pada pintu panel. Tampilan pilihan menu ini akan terus aktif sebelum sistem di *reset* melalui *smartphone* Android dengan memilih tombol "*RESET*" pada aplikasi Android. Seperti pada **Gambar 4.8.**

Gambar 4.8 Tampilan saat Sistem Aktif oleh Android

Pada **Gambar 4.8** menunjukkan tampilan GLCD setelah diaktifkan melalui *smartphone* yang digunakan untuk melakukan pengamanan lebih baik. Tampilan GLCD tersebut menunjukkan pilihan menu untuk melakukan eksekusi sistem dengan penjelasan sebagi berikut:

A – Daftar : Menu ini digunakan untuk melakukan proses *enrollment* atau pendaftaran sidik jari dengan memasukkan ID yang diinginkan dan mengikuti perintah meletakkan jari sesuai dengan yang diinginkan.

B – Hapus : Menu ini digunakan untuk melakukan proses penghapusan sidik jari yang telah didaftarkan dengan memasukkan ID yang ingin dihapus dan menekan "#" pada *keypad* sebagai fungsi *enter*. Jika berhasil terhapus, GLCD akan menampilkan tulisan "Berhasil Terhapus".

C – *Log In* : Menu ini digunakan untuk melakukan proses *log in* dengan meletakkan jari pada *scan area* pada *fingerprint scanner* dengan *delay* 2 *second* untuk mencocokkan sidik jari. Jika sidik jari terdaftar dan berhasil *log in*, maka GLCD akan menampilkan tulisan "Berhasil LOGIN". Jika sidik jari belum terdaftar dan tidak berhasil *log in*, maka GLCD akan menampilkan *page* pilihan menu untuk melakukan eksekusi kembali.

D – Hapus Data : Menu ini digunakan untuk menghapus data EEPROM. Jika berhasil, maka GLCD akan menampilkan tulisan "Menghapus EEPROM".

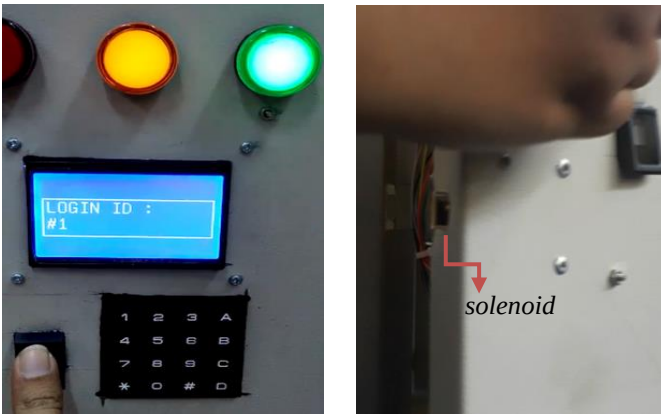
Gambar 4.9 Melakukan *Log In* dengan Masukkan Tombol C

Setelah diaktifkan melalui *smartphone* dan memunculkan tampilan pilihan pada GLCD maka dipilihlah tombol C untuk melakukan *Log In* seperti pada **Gambar 4.9.** Selanjutnya GLCD menampilkan seperti pada **Gambar 4.10** yaitu tertulis "*Login:* " dan posisi lampu *scan area* pada *fingerprint* menyala. Hal ini dikarenakan lampu tersebut berfungsi memperjelas proses pengambilan gambar sidik jari.

Gambar 4.10 Tampilan saat *Log In*

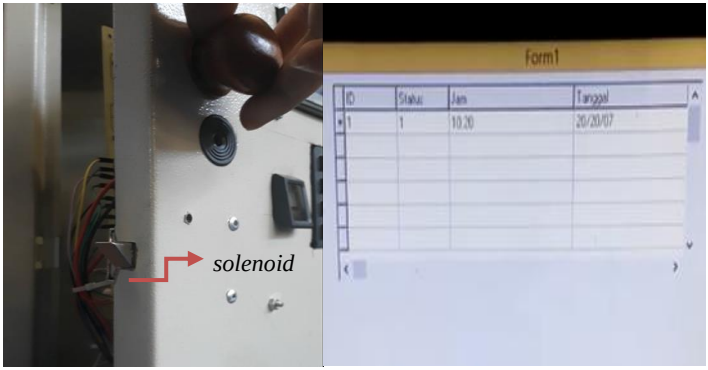
Pada **Gambar 4.11** menunjukkan kondisi GLCD yang menampilkan tulisan "*Login* ID: #1" yang berarti bahwa *Log In* berhasil dan ID yang terbaca yaitu 1. Pada saat sidik jari berhasil melakukan *Log*

In, maka lampu indikator hijau akan menyala dan *solenoid* akan aktif dengan posisi tuas kedalam. Lampu Indikator hijau dan *solenoid* akan aktif selama 4 *second*.



Gambar 4.11 Tampilan GLCD dan Kondisi *Solenoid* saat *Log In*

Setelah lampu Indikator hijau dan *solenoid* aktif selama 4 *second*, maka lampu dan *solenoid* tidak aktif dan mengirimkan informasi *Log In* kepada *software* Delphi 7 melalui kabel serial yang terhubung antara Arduino dan PC. Kondisi *solenoid* dan hasil Delphi 7 seperti pada **Gambar 4.12**.



Gambar 4.12 Kondisi *Solenoid* Setelah 4 *Second*

BAB V
PENUTUP

5.1 Kesimpulan

Setelah dilakukan pengujian dari masing masing komponen dan sistem secara keseluruhan oleh peneliti, maka dapat disimpulkan bahwa Sistem *doorlock* ini dapat bekerja sesuai dengan funginya sebagai pengaman dan *data logger* dengan keakuratan pembacaan sensor 99% berdasarkan pengujian *error* dengan ID yang sama dan menunjukkan bahwa terjadi kesalahan *log in* pada pengujian ke – 16.

5.2 Saran

Sistem mikrokontroler untuk membuka pintu panel dengan *password* ini telah berjalan sesuai dengan rancangannya, tetapi untuk menyempurnakan hasilnya perlu dipertimbangkan hal – hal sebagai berikut :

1. Perlu ditingkatkan proses komunikasi data dan *software* penampil data agar lebih mudah diakses.
2. Menambahkan tampilan perintah hapus EEPROM dan menampilkan rekapan *database* melalui android.
3. Menambahkan perintah *keypad* untuk membuka sistem utama pada GLCD,

-----Halaman ini sengaja dikosongkan-----

DAFTAR PUSTAKA

[1] Gayung A., "Sistem Pengaman Rumah dengan *Security Password* Menggunakan Sensor Gerak Berbasis Mikrokontroler AT89S51", *Tugas Akhir*, Universitas Sumatera Utara, Medan, 2009.

[2] Darmawiguna, I. G. M., Sunarya, I. M. G., "Aplikasi *Motion Detection* untuk *Home Security* Sistem dengan Pelaporan Otomatis Berbasis SMS *Gateway*", *Kumpulan Artikel Mahasiswa Pendidikan Teknik Informatika (KARMATI)*, Universitas Pendidikan Ganesha, Singaraja, 2013.

[3] Dwirani, E. I., "Implementasi Sidik Jari untuk Sistem Absensi pada SDN II Cineam Tasikmalaya", *Skripsi*, Universitas Komputer Indonesia, Bandung, 2004.

[4] Suroto, "Studi Penyempurnaan Identifikasi Sidik Jari pada *Algoritma Minutea"*, *Skripsi*, Universitas Indonesia, Depok, 2009.

----Halaman ini Sengaja Dikosongkan----

LAMPIRAN

1. Program Enrollment Sidik Jari

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
uint8_t id;
uint8_t getFingerprintEnroll(); // jenis unsigned integer panjang 8 bit

// Software serial for when you dont have a hardware serial port
// pin #2 is IN from sensor (GREEN wire)
// pin #3 is OUT from arduino  (WHITE wire)
// On Leonardo/Micro/Yun, use pins 8 & 9. On Mega, just grab a hardware serialport
//SoftwareSerial mySerial(15, 14);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial3);
// On Leonardo/Micro or others with hardware serial, use those! #0 is green wire, #1 is white
//Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial1);

void setup()
{ while (!Serial);  // For Yun/Leo/Micro/Zero/...
  delay(500);
  Serial.begin(9600);
  Serial.println("Adafruit Fingerprint sensor enrollment");

  // set the data rate for the sensor serial port
  finger.begin(57600);
  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1);
  }
}
uint8_t readnumber(void) {
  uint8_t num = 0;
  boolean validnum = false;
  while (1) {
```

```
    while (! Serial.available());
    char c = Serial.read();
    if (isdigit(c)) {
       num *= 10;
       num += c - '0';
       validnum = true;
    } else if (validnum) {
      return num;
    }
  }
}

void loop() // run over and over again
{
  Serial.println("Ready to enroll a fingerprint! Please Type in the ID # you want to save this finger as...");
  id = readnumber();
  Serial.print("Enrolling ID #");
  Serial.println(id);
  while (!  getFingerprintEnroll() );
}

uint8_t getFingerprintEnroll() {
  int p = -1;
  Serial.print("Waiting for valid finger to enroll as #"); Serial.println(id);
  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
    case FINGERPRINT_OK:
      Serial.println("Image taken");
      break;
    case FINGERPRINT_NOFINGER:
      Serial.println(".");
      break;
    case FINGERPRINT_PACKETRECIEVEERR:
      Serial.println("Communication error");
      break;
    case FINGERPRINT_IMAGEFAIL:
      Serial.println("Imaging error");
```

```
    break;
  default:
    Serial.println("Unknown error");
    break;
  }
}

// OK success!
p = finger.image2Tz(1);
switch (p) {
  case FINGERPRINT_OK:
    Serial.println("Image converted");
    break;
  case FINGERPRINT_IMAGEMESS:
    Serial.println("Image too messy");
    return p;
  case FINGERPRINT_PACKETRECIEVEERR:
    Serial.println("Communication error");
    return p;
  case FINGERPRINT_FEATUREFAIL:
    Serial.println("Could not find fingerprint features");
    return p;
  case FINGERPRINT_INVALIDIMAGE:
    Serial.println("Could not find fingerprint features");
    return p;
  default:
    Serial.println("Unknown error");
    return p;
}

Serial.println("Remove finger");
delay(2000);
p = 0;
while (p != FINGERPRINT_NOFINGER) {
  p = finger.getImage();
}
Serial.print("ID "); Serial.println(id);
p = -1;
Serial.println("Place same finger again");
```

```
while (p != FINGERPRINT_OK) {
  p = finger.getImage();
  switch (p) {
  case FINGERPRINT_OK:
    Serial.println("Image taken");
    break;
  case FINGERPRINT_NOFINGER:
    Serial.print(".");
    break;
  case FINGERPRINT_PACKETRECIEVEERR:
    Serial.println("Communication error");
    break;
  case FINGERPRINT_IMAGEFAIL:
    Serial.println("Imaging error");
    break;
  default:
    Serial.println("Unknown error");
    break;
  }
}

// OK success!
p = finger.image2Tz(2);
switch (p) {
  case FINGERPRINT_OK:
    Serial.println("Image converted");
    break;
  case FINGERPRINT_IMAGEMESS:
    Serial.println("Image too messy");
    return p;
  case FINGERPRINT_PACKETRECIEVEERR:
    Serial.println("Communication error");
    return p;
  case FINGERPRINT_FEATUREFAIL:
    Serial.println("Could not find fingerprint features");
    return p;
  case FINGERPRINT_INVALIDIMAGE:
    Serial.println("Could not find fingerprint features");
    return p;
```

```
    default:
      Serial.println("Unknown error");
      return p;
  }

  // OK converted!
  Serial.print("Creating model for #");  Serial.println(id);
  p = finger.createModel();
  if (p == FINGERPRINT_OK) {
    Serial.println("Prints matched!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_ENROLLMISMATCH) {
    Serial.println("Fingerprints did not match");
    return p;
  } else {
    Serial.println("Unknown error");
    return p;
  }
  Serial.print("ID "); Serial.println(id);
  p = finger.storeModel(id);
  if (p == FINGERPRINT_OK) {
    Serial.println("Stored!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_BADLOCATION) {
    Serial.println("Could not store in that location");
    return p;
  } else if (p == FINGERPRINT_FLASHERR) {
    Serial.println("Error writing to flash");
    return p;
  } else {
    Serial.println("Unknown error");
    return p;
  }
}
```

2. Program Scanning Sidik Jari

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
int getFingerprintIDez();

// pin #2 is IN from sensor (GREEN wire)
// pin #3 is OUT from arduino  (WHITE wire)
//SoftwareSerial mySerial(2, 3);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial3);
// On Leonardo/Micro or others with hardware serial, use those! #0 is green wire, #1 is white
//Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial1);

void setup()
{
  while (!Serial);  // For Yun/Leo/Micro/Zero/...
  Serial.begin(9600);
  Serial.println("Adafruit finger detect test");

  // set the data rate for the sensor serial port
  finger.begin(57600);
  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1);
  }
  Serial.println("Waiting for valid finger...");
}

void loop()                     // run over and over again
{
  getFingerprintIDez();
  digitalWrite(12, HIGH);
  delay(50);            //don't ned to run this at full speed.
}
uint8_t getFingerprintID() {
  uint8_t p = finger.getImage();
```

```
switch (p) {
  case FINGERPRINT_OK:
    Serial.println("Image taken");
    break;
  case FINGERPRINT_NOFINGER:
    Serial.println("No finger detected");
    return p;
  case FINGERPRINT_PACKETRECIEVEERR:
    Serial.println("Communication error");
    return p;
  case FINGERPRINT_IMAGEFAIL:
    Serial.println("Imaging error");
    return p;
  default:
    Serial.println("Unknown error");
    return p;
}

// OK success!
p = finger.image2Tz();
switch (p) {
  case FINGERPRINT_OK:
    Serial.println("Image converted");
    break;
  case FINGERPRINT_IMAGEMESS:
    Serial.println("Image too messy");
    return p;
  case FINGERPRINT_PACKETRECIEVEERR:
    Serial.println("Communication error");
    return p;
  case FINGERPRINT_FEATUREFAIL:
    Serial.println("Could not find fingerprint features");
    return p;
  case FINGERPRINT_INVALIDIMAGE:
    Serial.println("Could not find fingerprint features");
    return p;
  default:
    Serial.println("Unknown error");
    return p; }
```

```
  // OK converted!
  p = finger.fingerFastSearch();
  if (p == FINGERPRINT_OK) {
    Serial.println("Found a print match!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_NOTFOUND) {
    Serial.println("Did not find a match");
    return p;
  } else {
    Serial.println("Unknown error");
    return p;
  }

  // found a match!
  Serial.print("Found ID #"); Serial.print(finger.fingerID);
  Serial.print(" with confidence of "); Serial.println(finger.confidence);
}
// returns -1 if failed, otherwise returns ID #
int getFingerprintIDez() {
  uint8_t p = finger.getImage();
  if (p != FINGERPRINT_OK)  return -1;
  p = finger.image2Tz();
  if (p != FINGERPRINT_OK)  return -1;
  p = finger.fingerFastSearch();
  if (p != FINGERPRINT_OK)  return -1;

  // found a match!
  digitalWrite(12,LOW); //set the red LED off
  digitalWrite(11,HIGH);//set the green LED on
  delay(1000);
  digitalWrite(11,LOW);//set the green LED off
  delay(1000);
  digitalWrite(12,HIGH);//set the red LED on
  digitalWrite(12,LOW);
  Serial.print("Found ID #"); Serial.print(finger.fingerID);
  Serial.print(" with confidence of "); Serial.println(finger.confidence);
  return finger.fingerID;}
```

3. Program Delete Sidik Jari

```
#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
uint8_t id;
uint8_t getFingerprintEnroll;

// Software serial for when you dont have a hardware serial port
// pin #2 is IN from sensor (GREEN wire)
// pin #3 is OUT from arduino  (WHITE wire)
// On Leonardo/Micro/Yun, use pins 8 & 9. On Mega, just grab a hardware serialport
//SoftwareSerial mySerial(15, 14);
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial3);
// On Leonardo/Micro or others with hardware serial, use those! #0 is green wire, #1 is white
//Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial1);

void setup()
{
  Serial.begin(9600);
  Serial.println("Delete Finger");

  // set the data rate for the sensor serial port
  finger.begin(57600);
  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  } else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1);
  }
}

void loop()                     // run over and over again
{
  while (!Serial);  // For Yun/Leo/Micro/Zero/...
  delay(500);
  Serial.println("Type in the ID # you want delete...");
  uint8_t id = 0;
  while (true) {
```

```
    while (! Serial.available());
    char c = Serial.read();
    if (! isdigit(c)) break;
    id *= 10;
    id += c - '0';
  }
  Serial.print("deleting ID #");
  Serial.println(id);
  deleteFingerprint(id);
}

uint8_t deleteFingerprint(uint8_t id) {
  uint8_t p = -1;
  p = finger.deleteModel(id);
  if (p == FINGERPRINT_OK) {
    Serial.println("Deleted!");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
    Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_BADLOCATION) {
    Serial.println("Could not delete in that location");
    return p;
  } else if (p == FINGERPRINT_FLASHERR) {
    Serial.println("Error writing to flash");
    return p;
  } else {
    Serial.print("Unknown error: 0x"); Serial.println(p, HEX);
    return p;
  }
}
```

4. Program Sistem Secara Keseluruhan

```
#include <SoftwareSerial.h>
#include <Adafruit_Fingerprint.h>
#include <Wire.h>
#include "U8glib.h"
#include "RTClib.h"
#define AT24C02 0x50  // Address with three address pins grounded.
U8GLIB_ST7920_128X64 u8g(37, 35, 36, U8G_PIN_NONE);
RTC_DS1307 rtc;
Adafruit_Fingerprint finger = Adafruit_Fingerprint(&Serial3);

int jam, menit, detik, tanggal, bulan, tahun;
byte logdata[9];
char str[100], X, v;
int str_index,readaddr=0;// Mengirim maksimal 100 karakter
int i;

void setup()
{
  Wire.begin();
  finger.begin(57600);
  Serial.begin(9600);
  Serial2.begin(9600);
  rtc.begin();
  //rtc.adjust(DateTime(F(__DATE__),F(__TIME__)));

  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);//buzzer
  pinMode(4, OUTPUT);//solenoid
  pinMode(7, OUTPUT);//hijau
  pinMode(52, OUTPUT);
  pinMode(53, OUTPUT);
  pinMode(33, OUTPUT);

  analogWrite(2, 220);
  digitalWrite(3, HIGH);//buzzer
  digitalWrite(4, HIGH);//solenoid
  digitalWrite(5, HIGH);//merah
```

```
  digitalWrite(7, HIGH);//hijau
  digitalWrite(52, HIGH);
  digitalWrite(53, HIGH);
  digitalWrite(33, LOW);
  delay(500);
  keypad_init();
  logdata[7] = 10; //KARAKTER NEWLINE
  logdata[8] = 0;

  // assign default color value
  if ( u8g.getMode() == U8G_MODE_R3G3B2 ) {
    u8g.setColorIndex(255);     // white
  }
  else if ( u8g.getMode() == U8G_MODE_GRAY2BIT ) {
    u8g.setColorIndex(3);         // max intensity
  }
  else if ( u8g.getMode() == U8G_MODE_BW ) {
    u8g.setColorIndex(1);         // pixel on
  }
  else if ( u8g.getMode() == U8G_MODE_HICOLOR ) {
    u8g.setHiColorByRGB(255,255,255);
  }
  u8g.setFont(u8g_font_unifont);
}

void loop()
{
  awal:
  int id, caddr, result;
  char c, buff[20], *temp;
  c = 0;

  showMenu(0);
   //Serial.println("selamat datang");
  str_index = 0; //reset index penerimaan karakter serial
  X=Serial2.read();
  if (X=='A')//jika tidak ada karakter masuk lewat serial
  {goto menu;}
  if (X=='B'){goto eeprom;}
```

```
  goto awal;
menu:
{
  X=Serial2.read();
  if (X == 'R'){goto awal;}
  DateTime now = rtc.now();
  c = 0;
  c = keypad();

  showMenu(1);
    switch(c) //c = karakter yang diinput melalui keypad
    {
      case 'A' :  enrollID(atoi(waitInput("ID Baru :")));
                  break;
      case 'B' :  deleteID(atoi(waitInput("Hapus ID :")));
                  break;
      case 'C' :  showMenu(2);
                  while((id=getID()) == -1) if(keypad() == 'D') break;
//menunggu sidik jari atau cancel menekan keypad D

                  //memasukkan parameter rtc ke dalam format penyimpanan
                  logdata[2] = now.hour();
                  logdata[3] = now.minute();
                  logdata[4] = now.day();
                  logdata[5] = now.month();
                  logdata[6] = now.year()%2000;

                  if(id >= 0) //jika sidik jari terdaftar
                  {
                    logdata[0] = 1;
                    logdata[1] = id;
                    itoa(id, buff, 10);

                    //tampilan glcd
                    u8g.firstPage();
                    do {
                      u8g.drawRFrame(0, 16, 128, 35, 0);
                      u8g.drawStr(3, 29, "LOGIN ID :");
                      u8g.drawStr(3, 44, "#");
```

```
    u8g.drawStr(11, 44, buff);
  } while( u8g.nextPage() );

  //nyalakan relay dan buzzer
  digitalWrite(3, LOW);
  delay(1000);
  digitalWrite(3, HIGH);
  digitalWrite(4, LOW);
  digitalWrite(5, LOW);
  delay(3000);
  digitalWrite(4, HIGH);
  digitalWrite(5, HIGH);
}
else //Jika sidik jari tak terdaftar atau error
{
  u8g.firstPage();
  do {
    u8g.drawRFrame(0,16,128, 35, 0);
    u8g.drawStr(3, 29, "LOGIN GAGAL !");
  } while( u8g.nextPage() );

  logdata[0] = 0;
  logdata[1] = 0;
}

Serial.print ('A');
for(i=0; i<=1; i++)
{EEPROMWrite(i, logdata[i]);

Serial.print(logdata[i]);} //menulis data ke eeprom
Serial.print ('B');
for( i=2; i<=3; i++)
{
  if(logdata[i]==0 || logdata[i]<=9)
  {
    Serial.print(0);
    Serial.print(logdata[i]);
  }
  else {
```

```
                Serial.print(logdata[i]);
               }

              }
              Serial.print ('C');
              for( i=4; i<=6; i++)
              {
               if(logdata[i]==0 || logdata[i]<=9)
               {
                Serial.print(0);
                Serial.print(logdata[i]);
               }
               else {
                Serial.print(logdata[i]);
               }
              }
              goto menu;
              break;
     case 'D' :  showMenu(3);
              for(int  i=0;  i<2047;  i++)  EEPROMWrite(i,  0);  //menulis
karakter 0 (null) di semua alamat eeprom = menghapus data
              break;
    }
  }
goto menu;

eeprom:
  readaddr = 0; v = -1;
    for(int i=readaddr; i<=readaddr+7; i++) //membaca 8 data eeprom
    {
      //membaca eeprom dengan alamat i
      Serial.println(EEPROMRead(i));
     }
    readaddr += 8; //untuk membaca 8 alamat berikutnya
goto awal;
}

char * waitInput(char * header) //proses input angka dari keypad
{
```

```
  boolean del = false;
  int index = 0;
  char buff[10];
  memset(buff, 0, 10); //reset char array buff

  drawInput(header, buff);
  while(1)
  {
    char c = keypad(); //input karakter keypad

    if(c >= '0' && c <= '9') //untuk karakter 0-9
    {
      if(del == true) { index++; del = false; } //algoritma untuk menghapus
karakter dalam array
      buff[index++] = c; //menyimpan karakter c ke dalam char array buff
      if(index >= 8) index = 8; //input tidak bisa melebihi 8 karakter

      drawInput(header, buff); //menampilkan hasil input ke glcd
    }
    else if(c == '*') //karakter * adalah untuk menghapus input
    {
      if(del == false) { index--; del = true; }  //algoritma untuk menghapus
karakter dalam array
      if(index <= 0) { index = 0; memset(buff, 0, 10); } //jika tidak ada
karakter lagi untuk dihapus
      buff[index--] = 0; //karakter terakhir diganti dengan karakter 0 (null),
dengan kata lain sama dengan dihapus

      drawInput(header, buff); //tampilkan ke glcd
    }
    else if(c == '#') break; //enter, keluar dari perulangan
  }
  return buff; //nilai balik buff
}

void drawInput(char * header, char str[]) //menampilkan data ke glcd
{
  u8g.firstPage();
  do {
```

```
   u8g.drawStr(0, 11, header);
   u8g.drawStr(0, 26, str);
  } while( u8g.nextPage() );
}

void showMenu(int menu) //tampilkan data ke glcd
{
  u8g.firstPage();
  do {
    drawMenu(menu);
       DateTime now = rtc.now();
  } while( u8g.nextPage() );
}

void drawMenu(int stat) //tampilkan menu ke glcd
{
  char c, buff[20], *temp;
     DateTime now = rtc.now();
  switch(stat)
  {
    case 0:
      u8g.drawRFrame(0,0,128, 64, 0);
      u8g.drawStr(7, 13, "SELAMAT DATANG");
      sprintf(buff,  "%02d:%02d:%02d  ",     now.hour(),  now.minute(),
now.second());
      u8g.drawStr(7, 44, buff);
      sprintf(buff,  "%02d:%02d:%02d  ",     now.day(),  now.month(),
now.year());
      u8g.drawStr(7, 59, buff);
      break;
    case 1:
      u8g.drawRFrame(0,0,128, 64, 0);
      u8g.drawStr(3, 13, "A - Daftar");
      u8g.drawStr(3, 29, "B - Hapus");
      u8g.drawStr(3, 44, "C - Login");
      u8g.drawStr(3, 59, "D - Hapus Data");
      break;
    case 2:
      u8g.drawRFrame(0,16,128, 35, 0);
```

```
    u8g.drawStr(3, 29, "Login :");
    break;
   case 3:
    u8g.drawRFrame(0,16,128, 35, 0);
    u8g.drawStr(3, 29, "Menghapus EEPROM...");
    break;
  }
}

void EEPROMWrite(byte wordaddress, byte data) //fungsi simpan eeprom
{
 Wire.beginTransmission(AT24C02);
 Wire.write(wordaddress);
 Wire.write(data);
 Wire.endTransmission();
 delay(5);
}

byte EEPROMRead(byte wordaddress) //fungsi baca eeprom
{
  byte result;

  Wire.beginTransmission(AT24C02);
  Wire.write(wordaddress);
  Wire.endTransmission();//RESTART OBLIGATORY
  Wire.requestFrom(AT24C02, 1);
  if(Wire.available()) result = Wire.read();
  Wire.endTransmission();
  return result;
}

int dataFindC(char c) //cari karakter di eeprom, nilai kembalinya berupa index alamat
{
  int address = 0;
  while(EEPROMRead(address) != c) address++; //terus membaca eeprom hingga karakter yang diinginkan terdeteksi
```

```
  return address;

}

uint8_t enrollID(int id)
{
  int p = -1;

  showMessage("Pindai Jari");

  while (p != FINGERPRINT_OK) //selama tidak terpindai sidik jari
  {
    p = finger.getImage();
    if(p == FINGERPRINT_OK) //jika terpindai sidik jari, keluar dari
perulangan
    {
      showMessage("Terpindai !");
      //Serial.println("Image taken");
    }
  }

  p = finger.image2Tz(1);
  if(p != FINGERPRINT_OK) //Jika sidik jari gagal terpindai, keluar
(error)
  {
    showMessage("Error !");
    //Serial.println("Error !");
    return p;
  }

  showMessage("Angkat Jari");
 // Serial.println("Remove finger");
  delay(2000);

  p = 0;
  while (p != FINGERPRINT_NOFINGER) p = finger.getImage();
//menunggu sidik jari untuk dipindai
 // Serial.print("ID "); Serial.println(id);
```

```
  p = -1;
  showMessage("Ulangi Pemindaian");
  while (p != FINGERPRINT_OK) //selama tidak ada sidik jari terpindai
  {
    p = finger.getImage();
    if(p == FINGERPRINT_OK) //jika ada sidik jari terpindai, keluar dari
perulangan
    {
      showMessage("Terpindai !");
     // Serial.println("Image taken");
    }
  }

  p = finger.image2Tz(2);
  if(p != FINGERPRINT_OK) //Jika id tidak terdeteksi, keluar (error)
  {
    return p;
    //Serial.println("Error !");
  }

  // OK converted!
  showMessage("Menganalisa");
  //Serial.print("Creating model for #");  Serial.println(id);

  p = finger.createModel();
  if (p == FINGERPRINT_OK)
  {
  //  Serial.println("Prints matched!");
    showMessage("Sidik Jari Cocok !");
  }
  else //Jika sidik jari tidak terdaftar, keluar (error)
  {
   // Serial.println("Error !");
    showMessage("Error !");
    return p;
  }

  p = finger.storeModel(id);
  if (p == FINGERPRINT_OK) //Jika id terdeteksi
```

```
  {
    //Serial.println("Stored!");
    showMessage("Berhasil Menambahkan !");
  }
  else
  {
   // Serial.println("Error !");
    showMessage("Error");
    return p;
  }
}

int getID()
{
  uint8_t p = finger.getImage();
  if (p != FINGERPRINT_OK) //Jika id tidak terdeteksi, keluar
  {
    //Serial.println("Error !");
    return -1;
  }

  p = finger.image2Tz();
  if (p != FINGERPRINT_OK) //Jika id tidak terdeteksi, keluar
  {
  //  Serial.println("Error !");
    return -1;
  }

  p = finger.fingerFastSearch();
  if (p != FINGERPRINT_OK) //Jika id tidak terdeteksi, keluar
  {
   // Serial.println("Error !");
    return -2;
  }

  // found a match!
  return finger.fingerID; //Keluar dengan nilai baik ID sidik jari
}
```

```
int deleteID(int id)
{
  uint8_t p = -1;

  p = finger.deleteModel(id); //delete ID

  if (p == FINGERPRINT_OK) //Jika id terdeteksi
  {
    //Serial.println("ID Deleted !");
    showMessage("ID telah dihapus!");
    return p;
  }
}

void showMessage(char * text) //tampilkan pesan ke glcd
{
  u8g.firstPage();
  do {
    u8g.drawRFrame(0,16,128, 35, 0);
    u8g.drawStr(3, 29, text);
  } while( u8g.nextPage() );
}
void keypad_init() //inisialisasi pin input dan output untuk keypad
{
  pinMode(A8, INPUT);
  pinMode(A9, INPUT);
  pinMode(A10, INPUT);
  pinMode(A11, INPUT);
  pinMode(A12, OUTPUT);
  pinMode(A13, OUTPUT);
  pinMode(A14, OUTPUT);
  pinMode(A15, OUTPUT);

  digitalWrite(A8, HIGH);
  digitalWrite(A9, HIGH);
  digitalWrite(A10, HIGH);
  digitalWrite(A11, HIGH);
  digitalWrite(A12, HIGH);
  digitalWrite(A13, HIGH);
```

```
  digitalWrite(A14, HIGH);
  digitalWrite(A15, HIGH);
}

char keypad() //membaca input pada keypad, nilai balik berupa 1 karakter
{
  char key = 0;
  digitalWrite(A15, LOW); delay(5); //scanning baris
  if(digitalRead(A11) == 0) key = '1'; //deteksi tombol dan seterusnya
  else if(digitalRead(A10) == 0) key = '4';
  else if(digitalRead(A9) == 0) key = '7';
  else if(digitalRead(A8) == 0) key = '*';
  digitalWrite(A15, HIGH); delay(5);

  digitalWrite(A14, LOW); delay(5); //scanning baris
  if(digitalRead(A11) == 0) key = '2'; //deteksi tombol dan seterusnya
  else if(digitalRead(A10) == 0) key = '5';
  else if(digitalRead(A9) == 0) key = '8';
  else if(digitalRead(A8) == 0) key = '0';
  digitalWrite(A14, HIGH); delay(5);

  digitalWrite(A13, LOW); delay(5); //scanning baris
  if(digitalRead(A11) == 0) key = '3'; //deteksi tombol dan seterusnya
  else if(digitalRead(A10) == 0) key = '6';
  else if(digitalRead(A9) == 0) key = '9';
  else if(digitalRead(A8) == 0) key = '#';
  digitalWrite(A13, HIGH); delay(5);

  digitalWrite(A12, LOW); delay(5); //scanning baris
  if(digitalRead(A11) == 0) key = 'A'; //deteksi tombol dan seterusnya
  else if(digitalRead(A10) == 0) key = 'B';
  else if(digitalRead(A9) == 0) key = 'C';
  else if(digitalRead(A8) == 0) key = 'D';
  digitalWrite(A12, HIGH); delay(5);
  if(key != 0) delay(200);
  return key; //nilai kembali berupa karakter
}
```

5. Program *Database* pada Delphi 7

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, CPort, Grids, DBGrids, DB, ADODB;

type
  TForm1 = class(TForm)
    ComPort1: TComPort;
    Timer1: TTimer;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    ADOConnection1: TADOConnection;
    ADOTable1: TADOTable;
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    procedure FormCreate(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Timer1Timer(Sender: TObject);
    procedure delay(lama:longint);
    procedure Edit1Change(Sender: TObject);
   private
     { Private declarations }
   public
     { Public declarations }
   end;

  var
    Form1: TForm1;

  implementation

  {$R *.dfm}
  procedure TFORM1.delay(lama:longint);
  var ref:longint;
  begin
    ref:=gettickcount;
    repeat application.processmessages;
    until ((gettickcount-ref)>=lama);
  end;

  procedure TForm1.FormCreate(Sender: TObject);
  begin
  Comport1.Open;
  end;
```

```
procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
Comport1.Close;

end;

procedure TForm1.Timer1Timer(Sender: TObject);
Var
sett,cek,cek1,cek2,cek3,cek4,cek5,cek6:String;
begin
    repeat
      Comport1.ReadStr(sett,1);
      delay(1);
    until sett = 'A';
    Comport1.ReadStr(cek,1);
    Comport1.ReadStr(cek1,1);
    repeat
      Comport1.ReadStr(sett,1);
      delay(1);
    until sett = 'B';
    Comport1.ReadStr(cek2,2);
    Comport1.ReadStr(cek3,2);
    repeat
      Comport1.ReadStr(sett,1);
      delay(1);
    until sett = 'C';
    Comport1.ReadStr(cek4,2);
    Comport1.ReadStr(cek5,2);
    Comport1.ReadStr(cek6,2);
    delay(10);
    edit1.Text:=cek;
    edit2.Text:=cek1;
    edit3.Text:=cek2+':'+cek3;
    edit4.Text:=cek4+'/'+cek4+'/'+cek5;
    ADOTABLE1.Insert;
    ADOTABLE1.FieldByName('ID').AsString := edit1.Text;
    ADOTABLE1.FieldByName('Status').AsString := edit2.Text;
    ADOTABLE1.FieldByName('Jam').AsString := edit3.Text;
    ADOTABLE1.FieldByName('Tanggal').AsString := edit4.Text;

end;

procedure TForm1.Edit1Change(Sender: TObject);
begin

end;

end.
```

--- Halaman ini sengaja dikosongkan ---

DAFTAR RIWAYAT PENULIS

Nama : Aswindha Nasrullah
TTL : Sidoarjo, 15 September 1995
Jenis Kelamin : Perempuan
Agama : Islam
Alamat : Perumahan Sidokare Indah Blok AI/21, Sidoarjo
Telp/HP : 087 857 499 299
E-mail : naswiindha@gmail.com

RIWAYAT PENDIDIKAN

1. 2002 – 2008 : SD Negeri Pucang IV Sidoarjo
2. 2008 – 2011 : SMP Negeri 5 Sidoarjo
3. 2011 – 2014 : SMA Negeri 4 Makassar
4. 2014 – 2017 : Departemen Teknik Elektro Otomasi, Program Studi Teknik Elektro Industri - Fakultas Vokasi - Institut Teknologi Sepuluh Nopember (ITS)

PENGALAMAN KERJA

1. Kerja Praktek di Delta Jaya Mas, Gresik
2. Kerja Praktek di PT. PLN (Persero) Area Surabaya Barat, Surabaya

PENGALAMAN ORGANISASI

1. Anggota Departemen KOMINFO BEM FTI-ITS
2. Anggota Departemen KOMINFO HIMAD3TEKTRO